Hung, Jane Y. and Gonzalez, Luis F. and Walker, Rodney A. and Periaux, Jacques (2009) *Mission optimisation and multi-disciplinary design of hybrid unmanned aerial systems (UAS) using advanced numerical techniques.* In: 3rd Australasian Unmanned Air Vehicles Conference, 9-12 March 2009, Melbourne Convention Centre, Melbourne.

Mission Optimisation and Multi-Disciplinary Design of Hybrid Unmanned Aerial Systems (UAS) using Advanced Numerical Techniques

Miss Jane Y. Hung, Dr. Luis F. González, and Professor Rodney A. Walker
Queensland University of Technology
Brisbane, Queensland, Australia
Phone: +61 – 7 – 3138 1362
Email: {y.hung, l.gonzalez, ra.walker}@qut.edu.au

and

Professor Jacques Périaux
International Center for Numerical Methods in Engineering (CIMNE), Universidad Politécnica de Cataluña
Barcelona, Spain
Email: jperiaux@gmail.com

Abstract
This paper describes the theory and practical application of Hierarchical Asynchronous Parallel Multi-objective Evolutionary Algorithms (HAPMOEA) for mission optimisation of Unmanned Aerial Systems (UAS). Optimisation has emerged as a new discipline for UAS in recent years and most of the optimisation efforts are focused on the use of gradient-based techniques. One drawback of these methods is that they are mostly suitable when there is only one objective to be met with or when the objectives are differentiable. A real design or simulation will have more than one objective such as minimising fuel consumption, drag or time to complete the mission. It is usually the case that the problem is highly non-linear and non-differentiable. New techniques are required, and one of such techniques, even though computationally more intensive than gradient-based methods, are Evolutionary Algorithms (EAs). This paper describes an advanced EA methodology and its coupling with simulation analysis tools. Results will indicate the practicality and robustness of the method in finding optimal solutions and Pareto trade-offs between fuel consumption and time to complete the mission of a hybrid UAS by producing a set of non-dominated trajectories and mission from which the designer can choose.

Keywords: Hybrid Unmanned Aerial Systems (UAS), Multi-Disciplinary Design and Simulation, Evolutionary Algorithms, Mission Optimisation.

Biography:

Miss Jane Hung
Jane Hung completed a Bachelor of Engineering (Aerospace Avionics) with Honours and a Bachelor of Applied Science (Mathematics) in 2003 at QUT. Jane is currently working towards a PhD in the field of multi-objective mission planning for UAS utilising hybrid-propulsion systems.

Dr. Luis F. González
Dr Gonzalez is a lecturer at ARCAA and QUT. Three years experience as a Mechanical and Project Engineer in CFD/FEA design, analysis and installation. He has developed and managed several electromechanical projects for metallurgic, aeronautical and mechanical design companies. He has published sixteen refereed conference papers and four journal papers on the topic of Unmanned Aerial Systems.

Professor Rodney A. Walker
Prof. Walker is a lecturer in the School of Engineering Systems at QUT and is the Director of the Australian Research Centre of Aerospace Automation. His areas of expertise include Aerospace Avionics, Satellite Navigation, General Aviation and Airspace Management. He has worked on high value real world projects such as FedSat in addition to educational and research roles specializing in GPS applications and UAV's.

Professor Jacques Périaux
Prof. Périaux is a senior adviser fellow of Dassault Aviation. He is also a visiting professor at the University of Jyväskylä and the founder and vice-president of European Community on Computational Methods in Applied Sciences (ECCOMAS) organization. His research interest is in the area of computational sciences. He has over 60 refereed publications.

Introduction

Unmanned Aerial Systems (UAS) are becoming important military and commercial assets for diverse applications, ranging from reconnaissance and surveillance, to aid relief and monitoring tasks [1]. These vehicles are now available in a broad size and capability range and are intended to fly in regions where the presence of onboard human pilots is either too risky or unnecessary. Civilian applications for UAS technology are quickly emerging as a large and lucrative new aerospace market. Examples of civilian applications include, to name a few, coastal surveillance, power-line inspection, traffic monitoring, bush-fire monitoring, precision farming and remote-sensing. The multi-physics aspects of these vehicles can benefit from alternative approaches for design and optimisation [2,3].

Robust Framework

The complex task of aircraft design is now assisted by highly sophisticated analysis tools such as computational fluid dynamics (CFD), finite element analysis (FEA) and mission simulation. The logical extension to this progress is undoubtedly optimisation. This activity has emerged as a new discipline and most of the aerodynamic and structural optimisation efforts focus on the use of gradient-based techniques. One drawback of these methods is that they are mostly suitable when there is only one objective to be met with or when the objectives are differentiable.

A real design of an UAS will have more than one objective such as minimising fuel consumption, drag, as well as maximising range and endurance. In order to find the optimal solutions, new techniques are required. One of such techniques, even though computationally more intensive than gradient-based methods, are Evolutionary Algorithms (EAs) [3-5].

An optimisation tool, the Hierarchal Asynchronous Parallel Multi-objective Evolutionary Algorithm tool was used in this research. As indicated by its acronym, HAPMOEA, this tool uses advanced concepts for the solution: a modified canonical evolution strategy [6,7], capabilities for multi-objective optimisation using a Pareto tournament selection [8], capabilities for parallel computation using a modified asynchronous approach [3,9] and a hierarchical/multi-fidelity approach for the solution [3,5]. HAPMEA has been compared and has shown some computational benefits to other EA methods [3, 19]. Other multi-objective approaches such as HEA [5], NSGA or PGA [8, 9] can be used in combination with the UAV model presented later in this paper.

Hierarchical/Multi-Fidelity Population Topology

A hierarchical/multi-fidelity population topology, when integrated into an evolution algorithm, means that a number of separate populations are established in a hierarchical layout to solve the given problem, rather than a single 'cure-all' type single population layout. This method was proposed by Sefrioui [5] and is shown in Figure 1. The bottom layer uses a simple model and can be entirely devoted to exploration, the intermediate layer is a compromise between exploitation and exploration, while the top layer uses a refined model and concentrates on promising solutions.

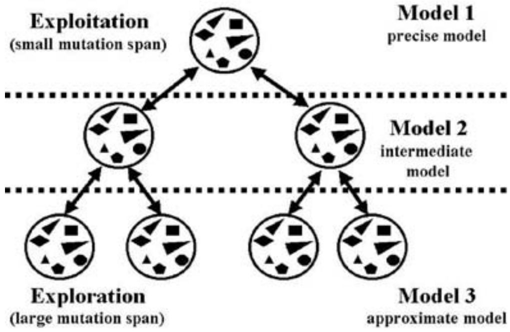

Figure 1: Hierarchical Topology

In simulation tools, the accuracy of the solution is often related to the sampling time, or how often each instance of the simulation cycle is calculated. The smaller the sampling time, the more accurate the simulation will be to real-time execution. However, a smaller sampling time implies a longer simulation time, due to limitations on the computational power available.

Applying the hierarchical/multi-fidelity topology to simulations, the top layer uses a precise time-consuming sampling time, whereas coarser sampling times are used in the intermediate and bottom layers, resulting in a more approximate model.

Parallel Computing

The algorithm used in this approach is a master-slave pMOEA but incorporates the concept of isolation and migration through hierarchical topology binary tree structure, where each level executes different MOEAs/parameters (heterogeneous). The parallel environment used is a cluster of PCs, wherein the master carries on the optimisation process while remote nodes compute the analysis solver environment. The message-passing model used is the Parallel Virtual Machine (PVM) [10]. A schematic of the parallelisation approach with asynchronous evaluation is shown in Figure 2. This algorithm has been tested in a cluster of heterogeneous CPUs, RAMs, caches, memory access times, storage capabilities and communication attributes. In this work, a cluster than can be configured with up to 18 machines with performances varying between 2.0 and 2.4GHz was used. Studies showing the performance of the algorithms are presented in Reference 19.

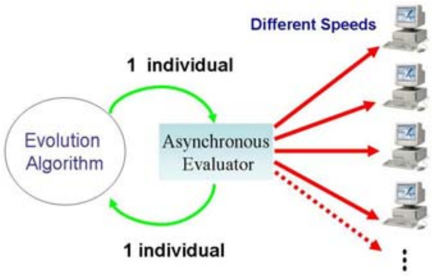

Figure 2: Asynchronous Evaluation

Asynchronous Solution

When considering the solution to a Multi-objective and Multidisciplinary Optimisation problem, several issues arise, as many methods of solution used in engineering today may take different times to complete their operation [7]. The classic example of this is a CFD, an FEA solver or a MATLAB® Simulink® simulation. With a typical industrial code used for simulation and analysis of aircraft, the time for the solution to converge to a specified level (either machine zero or an arbitrarily selected higher value) can vary over a significant range. The time taken for an iterative solution of non-linear partial differential equations is strongly dependent upon geometry or trajectory flown. The previous generation of EAs have mostly used a generation-based approach and so are the traditional genetic algorithm and evolution strategy. A difficulty with generational models is that they create an unnecessary bottleneck when used on parallel computers. If the population size is approximately equal to the number of processors, and most of the candidate off springs that are sent for solution can be successfully evaluated, then some processors will complete their task quickly with the remainder taking more time. With a generational approach, those processors that have already completed their solutions will remain idle until all processors have completed their work [6].

The approach used here is to ignore any concept of a generation-based solution. This approach is similar to the work by Wakunda and Zell [9] and other non-generational approaches. However, the selection operator is quite different, as it couples one-by-one (steady-state) function evaluation with a direct multi-objective fitness criterion. Whilst a parent population exists, offspring are not sent as a complete 'block' space to the parallel slaves for solution. Instead, one candidate is generated at a time, and is sent to any idle processor where it is evaluated at its own speed. When candidates have been evaluated, they are returned to the optimiser and either accepted by insertion into the main population or rejected. This requires a new selection operator because one offspring cannot now be compared against another, or even against the main population due to the variable-time evaluation. To overcome this, the recently evaluated offspring was compared against a previously established rolling-benchmark and, if successful, it replaces, according to some rule, a pre-existing individual in the population. This benchmarking is implemented via a separate evaluation buffer B, which provides a statistical 'background check' on the comparative fitness of the solution. The length of the buffer should represent a reasonable statistical sample size, but need not be too large; approximately twice the population size is more than adequate. When an individual has had a fitness assigned, it is then compared to past individuals (both accepted and rejected) to determine whether or not it should be inserted into the main population. If it is to be accepted, then some replacement strategy is invoked and it replaces a member of the main population. The replace-worst-always method is used exclusively in this work.

Multi-Objective Optimisation

Most evolutionary algorithms configured for multi-objective optimisation currently use the non-dominated sorting approach. This is a straightforward way to adapt an algorithm that is designed as a single objective optimiser into a multi-objective optimiser, and is used by many researchers [8]. The problem with sorting approaches is that the method is not a fully integrated one. Briefly, a sorting method works by computing the set of non-dominated solutions amongst a large statistical sampling (either a large population or previous data), and assigning these solutions a rank one. Then ignoring these points, the process is repeated until a 'second' Pareto front is found, and this is assigned a rank two. This process continues until all points are ranked, and then the value of the rank is assigned to the individual as a new single objective fitness. A problem arises now on whether it is fair to assign individuals in the second rank numerically half the fitness of the first, and whether the third rank deserves a third of the fitness of the first. This poses a dilemma regarding the level of equality present amongst the solutions, as often solutions with excellent information may lie adjacent to, but not in, rank one. To solve this 'artificial scaling' problem, it is possible to introduce scaling, sharing and niching schemes, however all of these require problem-specific parameters or knowledge, even in adaptive approaches. It is of course always desirable to compose an algorithm that does not introduce such unnecessary parameters.

The on-the-fly selection operator was implemented by means of a Pareto tournament selection operator. To implement an optimisation algorithm that is equally applicable to both single- and multi-objective problems, a suitable selection operator capable of handling either situation must be developed. An extension of the standard tournament operator popular in many approaches [8] was proposed.

The current operator is a novel approach in that it requires no additional 'tuning' parameters, works seamlessly with the asynchronous selection buffer B, and is very easy to encode. Simply, to determine whether a new individual x is to be accepted into the main population, it is compared with the selection buffer by assembling a small subset of the buffer called the t tournament functions as follows:

$$Q = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} \quad (1)$$

Q is assembled by selecting individuals from the buffer, exclusively at random, until it is full. Then it is ensured that that the new individual is not dominated by any in the tournament. If this is the case, then it is immediately accepted, and is inserted according to the replacement rules. The only parameter that needs to be determined in advance is the tournament size, a parameter that would exist in a single-objective optimisation anyway. Selection of this parameter requires a small amount of problem specific knowledge, and should vary between $Q = \frac{1}{2}B$ (strong selective pressure) and $Q = \frac{1}{6}B$ (weak selective pressure). The optimiser is not overly sensitive to this value, provided the user errs on the side of weak selective pressure (smaller tournaments) in the absence of better information. The egalitarian approach to the tournament, by selecting individuals at random, ensures good diversity amongst the selected individuals; no niching or forced separation of individuals has been found to be necessary. It can also be seen that in the event that the fitness vectors have only one element (a single-objective optimisation), this operator simplifies to the standard tournament selection operator [8].

UAV Simulation

To validate the effectiveness of HAPMOEA in the area of mission planning for UAS, it is necessary to demonstrate the improvements on the mission objectives in comparison to existing optimisation methods, namely gradient based methods. Simulations are used for this purpose as actual flight tests are often extremely time-consuming as well as cost-prohibitive.

The simulation consists of two major components: aircraft simulation model and mission scenarios. These will be described in the following sections.

Aircraft Simulation Model

The computer simulations for validating HAPMOEA in UAS mission planning were conducted in the MATLAB® Simulink® simulation environment combined with the AeroSim Blockset [11], which offers a comprehensive aircraft simulation and analysis package [12]. The AeroSim Blockset also offers a detailed model of an Aerosonde™ UAV, a real-world UAS, with a complete set of parameter to simulate the Aerosonde™ in flight. A modified version of this model was utilised in the construction of the aircraft simulation model. The entire simulation model is shown below in Figure 3, which has the addition of Aircraft Control and Flight Planner modules to the original Aerosonde™ model for unmanned operations.

The Aerosonde UAV block contains a detailed model of an Aerosonde™ UAV, which consists of Aerodynamic, Propulsion, Atmosphere, Aircraft Inertia, Acceleration and Moments, Equations of Motion and Earth models. The inputs into the Aerosonde UAV block are the aircraft control inputs (control surfaces, throttle, mixture ignition), and wind velocities. The outputs of the block are the aircraft states in various coordinate systems and aircraft coefficients. Many of these output values are used in the calculations of other blocks in the simulation model, while some are displayed for reference purposes, passed to FlightGear for visualisation through the FlightGear 0.9.2 Interface, or passed to the MATLAB® for further computations.

The Flight Planner block, developed at QUT, calls a MATLAB® function which calculates the necessary bearing/yaw adjustment from the current position of the aircraft to reach a desired waypoint according to a pre-specified list of waypoints. The waypoints are listed in the latitude, longitude and altitude coordinates. This bearing/yaw adjustment, along with other parameters, are passed to the Flight Control block.

The Flight Control block obtains outputs from the Flight Planner block and using the PID and PI controllers residing in this block, the values for aircraft controls are computed. These values form the control inputs to the Aerosonde UAV block to complete the loop for unmanned aircraft operations.

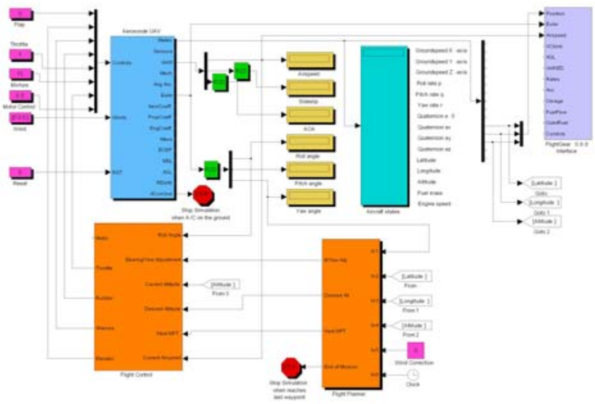

Figure 3: Aircraft Simulation Model

The above aircraft simulation model has not yet taken into consideration the effects of weather elements such as wind on the performance of the aircraft during flight. This will be included in future revisions of the aircraft simulation model.

Mission Scenario

In order to examine the effectiveness of HAPMOEA in UAS mission planning when compared to that of gradient-based methods, a baseline mission scenario was constructed. This mission scenario includes basic UAS operations such as Climb, Cruise, Descent and Loiter, and follows the mission profile in Figure 4.

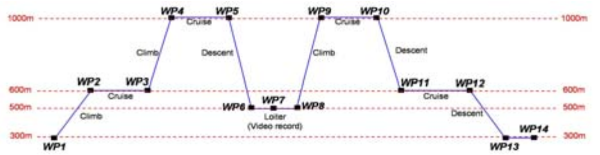

Figure 4: Mission Profile (not to scale)

A long-distance mission scenario with a total distance of approximately 400km has been constructed to clearly observe the comparisons in the optimization objectives between HAPMOEA and gradient-based methods. This realistic mission scenario utilises GPS waypoints located in central Queensland, Australia, indicated below in Figure 5.

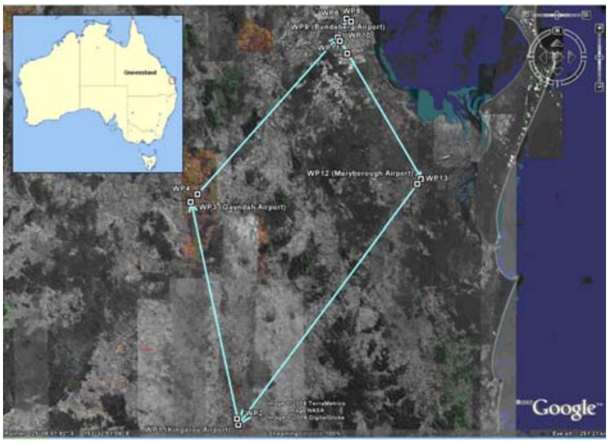

Figure 5: Baseline Mission Scenario with GPS Waypoints in Central Queensland, Australia (image generated using Google™ Earth)

Application: UAS Mission Trajectory Optimisation

Single-Objective Optimisation: Problem Definition

This single-objective problem considers the trajectory optimisation for an UAS mission. The objective considered is the minimisation of fuel consumption. The baseline flight mission for the UAS is defined using the GPS waypoints shown in Figure 5. However, the waypoints which the UAS has to pass through are not definite and may be adapted to achieve the optimisation objectives defined above. An example of such adaptation of mission waypoints is demonstrated below in Figure 6.

It needs to be noted that the leg from Waypoints 6 to 8 in the mission is the 'loiter' phase, as can be seen from Figure 4, during which some special mission requirements are carried out. Therefore, it is desired that these waypoints remain as specified and not be involved in the optimisation process.

Single-Objective Optimisation: Definition of Design Variables

The design variables used for UAS mission trajectory optimisation are the locations of the waypoints. The flight mission is made up of a series of waypoints and each waypoint is defined by its coordinates: latitude, longitude and altitude. The entire mission will be passed through the optimisation process in order to determine a set of waypoints which will achieve the optimisation objectives.

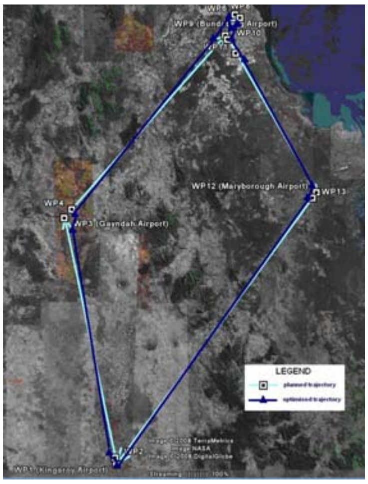

Figure 6: Example Comparison of an Optimised Trajectory (navy) for Minimum Fuel and Flight Time to the Baseline Flight Mission (light blue) (not to scale)

In this paper, South latitudes and East longitudes are taken as positive values, while South and East as negative. This convention was chosen for convenience as the waypoints considered for the simulation are located in Australia. Also, the latitudes and longitudes are measured in radians, and the altitudes in metres.

Single-Objective Optimisation: Definition of Fitness Objective Function

The fitness function is defined as the minimisation of fuel consumption, FC :

$$\min\left(f_1\right):\ f_1 = FC \qquad (2)$$

Single-Objective Optimisation: Definition of Constraints

The process of optimising the trajectory of a UAS mission is implemented taking into consideration of a number of constraints, namely the upper and lower bounds of waypoint coordinates and physical constraints.

Upper and Lower Bounds of a Waypoint

The upper and lower bounds of a waypoint defines a set of values from which a candidate waypoint, the possible coordinates to be passed through during the mission, is selected. The generation of a set of candidate waypoints forms a key component of the optimisation process. These upper and lower bounds of each coordinate – namely latitude, longitude and altitude – are defined taking into consideration of mission requirements as well as airspace and class restrictions. The chosen set of waypoints is then used in the 'solver' component of the optimisation process to determine the fitness function of this particular set of waypoints.

The upper and lower bounds for altitude, in metres, is calculated simply using a specified altitude margin, Δalt. In this paper, the value of Δalt was chosen to be 100m.

On the other hand, defining the bounds for latitude and longitude of a waypoint is more complex. This is due to the fact that realistically when the aircraft is flying, the distance between two waypoints is not the straight line route, but the line vertically above the straight line route following the earth's surface, known as a 'great circle'. Thefore spherical-triangle calculation is used for range and bearing related computations along a great circle [13], such as those required for defining the bounds for latitude and longitude of a waypoint. In this paper, the upper and lower bounds of the latitude and longitude are defined and calculated as 10% of the distance between the waypoint and the preceding waypoint, found using a rearranged form of the Haversine Formula [14].

Physical Constraints

Several physical constraints were incorporated into the aircraft simulation model, shown in Figure 3, with the intention of making the model more realistic. The majority of these constraints were implemented as part of the Flight Control block of the simulation model and these are:

- The throttle control must be within a range from 1% to 100% ($0.01 \le TC \le 1$).
- The rudder deflection must be between -20° and +20° ($-20° \le \delta_r \le 20°$).
- The aileron deflection must be between -10° and +10° ($-10° \le \delta_a \le 10°$).
- The elevator deflection must be between -20° and +20° ($-20° \le \delta_e \le 20°$).
- The airspeed is controlled at 20m/s when the aircraft at level flight and climbing, and increased to 30m/s when descending.

All these constraints are applied within the aircraft simulation model and are part of the simulation process.

Single-Objective Optimisation: Implementation Design and Optimisation Rationale

In the implementation of the UAS trajectory optimisation, two approaches were used. The first approach utilises a gradient-based optimisation strategy which is an in-built function in MATLAB®, and the other approach uses the EA strategy which is inherent in the HAPMOEA optimiser. Both approaches use the MATLAB® aircraft simulation model as shown in Figure 3, with the fundamental sample time, Δt, set at 0.1 seconds. Each of these approaches is described below.

Gradient-Based Optimisation Strategy

The gradient-based optimisation approach utilises the in-built constrained solver, which finds a minimum of a constrained nonlinear multivariable function, which is defined as follows:

Find $\min_x f(x)$ subject to:

$$\begin{gathered} c(x) \le 0 \\ c_{eq}(x) = 0 \\ A \cdot x \le b \\ A_{eq} \cdot x = b_{eq} \\ lb \le x \le ub \end{gathered}$$

where x is the vector input to the function $f(x)$, b and b_{eq} are the vector bounds for linear equalities, lb and ub are upper and lower bounds for the design variables, A and A_{eq} are coefficient matrices, $c(x)$ and $c_{eq}(x)$ are functions that return vectors, and $f(x)$ is a function that returns a scalar. $f(x)$, $c(x)$ and $ceq(x)$ can be nonlinear functions.

The optimisation process requires the specification of an initial estimate vector, x_0, and a set of constraints for the variables to be optimised. The constraints can be any or a combination of the following:

1. Linear equalities ($A \cdot x \le b$ and $A_{eq} \cdot x = b_{eq}$)
2. Lower and upper bounds ($lb \le x \le ub$)
3. Nonlinear inequalities ($c(x) \le 0$ or $c_{eq}(x) = 0$)

In the case of the UAS trajectory optimisation problem, only the lower and upper bounds are used in the setup of the constraint solver.

The optimisation algorithm used by the constraint solver can be one of two types: large-scale or medium-scale. The large-scale algorithm employs a subspace trust-region method and is based on an interior-reflective Newton method, while the medium-scale optimisation uses a sequential quadratic programming (SQP) method [15-18]. For the implementation of the UAS trajectory optimisation problem, the SQP method is used since the large-scale method does not currently solve the type of problem which the UAS trajectory optimisation problem poses.

The setup of the SQP solver for the case of the UAS trajectory optimisation problem consists the definition of the initial estimates vector, x_0, and the lower and upper bounds vectors, lb and ub, respectively. The initial estimates vector, x_0, is defined as a column vector as follows:

$$x_0 = \begin{bmatrix} lat_1 \\ lon_1 \\ alt_1 \\ \vdots \\ lat_n \\ lon_n \\ alt_n \end{bmatrix} = \begin{bmatrix} 0.4635 \\ 2.6500 \\ 600 \\ \vdots \\ 0.4639 \\ 2.6501 \\ 300 \end{bmatrix} \qquad (3)$$

where lat_i, lon_i and alt_i are the latitude, longitude and altitude coordinates of the ith waypoint respectively, in a mission with n waypoints. The values of the waypoint coordinates are obtained from the baseline mission waypoints as shown in Figure 5. The lower and upper bounds vectors, lb and ub, respectively, are calculated as per §"Single-Objective Optimisation: Definition of Constraints - Upper and Lower Bounds of a Waypoint". These are used by the SQP solver to generate candidate waypoint coordinates which is then passed to the function to be minimised.

In the case of the UAS trajectory optimisation problem, the function to be minimised calls the simulation model, which accepts the vector of the candidate waypoint coordinates. The function then constructs a waypoint table from this column vector as a $n \times 3$ matrix. This set of candidate waypoints is then used by the aircraft simulation model to execute the mission. The output of the simulation is the fuel consumption during the mission, which is the objective to be minimised by the SQP solver.

EA-Based Optimisation Strategy

The EA approach to the UAS trajectory optimisation problem is set up with only one layer. This layer has a population size of 10, two parents in recombination, a buffer length of 12 and a tournament-in-buffer ratio of 2.0 (refer to §"Robust Framework").

The optimisation rationale is display below in Figure 7.

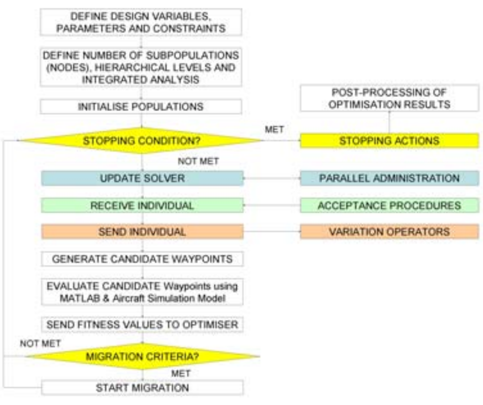

Figure 7: The EA Optimisation Rationale Flow Diagram

The EA optimiser calls the MATLAB® aircraft simulation model in order to evaluate each candidate waypoints which were generated in the process. The output of the aircraft simulation model is the fuel consumption over the mission and is the objective which is to be minimised by the EA optimiser.

Note that an initial estimates vector is not necessary for the EA optimiser.

Single-Objective Optimisation: Optimisation Results and Post-Processing of Optimal Solutions

The optimisation procedure for each approach used different stopping conditions. For the gradient based approach, the default stopping condition of the SQP solver was used, which terminates the process if the magnitude of the directional derivative in the search direction is less than 2×10^{-6} and the maximum constraint violation is less than 1×10^{-6}. On the other hand, the EA algorithm was run for 500 evaluations on one machine only. Figure 8 and Figure 9 illustrate the fitness-vs-function evaluations graphs for both approaches in the optimisation of one waypoint.

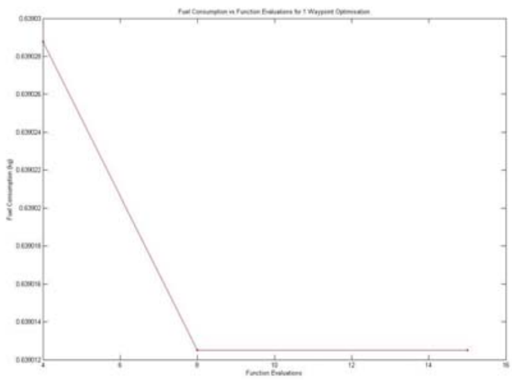

Figure 8: Fuel Consumption vs Function Evaluations (1-Waypoint Optimisation – SQP)

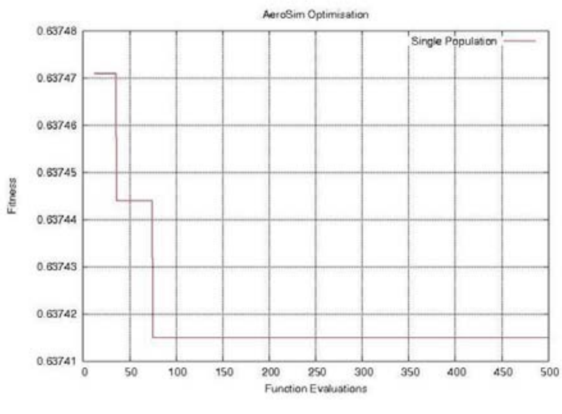

Figure 9: Fuel Consumption vs Function Evaluations (1-Waypoint Optimisation – EA)

Table 1 compares the optimisation parameters, the run time statistics and the fitness values when using the two approaches. The fitness values are also compared to that of the baseline mission, which has a fuel consumption value of 0.7kg.

Table 1: Table of Statistics for Single-Objective Optimisation

Optimisation Method	*SQP*	*EA*
# Waypoints/Design Variables Optimised	1 (3 DV)	1 (3 DV)
Function Evaluations Performed	15	500
Time Taken (hrs)	19.6	112.8
Fuel Consumption (kg)	0.63901	0.637475
% Improvement from Baseline Mission	8.7%	8.9%

It can be seen that whereas the EA approach is more computationally expensive, it gives a set of waypoints which execute to give a lower value of fuel consumption. The difference in the fitness values generated will increase as the number of waypoints to be optimised increases. Also, the use of an EA strategy will provide additional advantages for Multi-Objective problems.

Multi-Objective Optimisation

This multi-objective problem considers the same trajectory optimisation for an UAS mission as the single-objective problem in §"Application: UAS Mission Trajectory Optimisation - Single-Objective Optimisation: Problem Definition", using the same design variables as per §"Application: UAS Mission Trajectory Optimisation - Single-Objective Optimisation: Definition of Design Variables". The objectives considered in this case are the minimisation of fuel consumption, FC, and the time required, T_{req}, to execute the mission as per Figure 5 and are evaluated using the following fitness functions:

$$\min\left(f_1\right):\ f_1 = FC \qquad (4)$$

$$\min\left(f_2\right):\ f_2 = T_{req} \qquad (5)$$

The constraints for the multi-objective optimisation are defined in the same way as that of the single-objective optimisation in §"Application: UAS Mission Trajectory Optimisation - Single-Objective Optimisation: Definition of Constraints".

The UAS trajectory optimisation problem was implemented using the EA approach only to demonstrate the capability of the EA optimiser to handle such a problem.

The EA approach for the multi-objective optimisation problem was executed with the same procedure as outlined in §"Robust Framework", with the exception of the two objectives to be minimised – fuel consumption, FC, and the time required to execute the mission, T_{req}.

The multi-objective optimisation was run for 96 hours on one machine only.

Table 2 shows the run time statistics and the fitness values for a selected number of Pareto front members.

Table 2: Table of Statistics for Multi-Objective Optimisation

# Pareto Front Members	Fuel Consumption (kg)	Mission Time Required (hrs)	Time Taken (hrs)
1 Pareto Front Member	0.703125	6.023	96

It can be seen that the use of a MOEA optimiser provides the design team the additional advantage of finding a solution to a multi-objective problem, such as one demonstrative above.

Conclusions

The basic concepts of a hierarchical, asynchronous parallel multi-objective evolutionary algorithm used to solve the problem of optimising an UAS mission were presented in this paper. Even though more computationally expensive, an EA optimiser can provide an UAS mission planner extended benefits in terms of improved endurance and/or range, and extra payload capacity. The method can be used as an alternative option to satisfy some of the needs for robust multi-objective and multidisciplinary design optimisation problems. The method is easily coupled, particularly adaptable, easily parallelised, and requires no gradient of the objective function(s). The methodology is integrated in a single framework that allows:

- Solving of single and multi-objective, non-linear, deceptive, discontinuous, and multi-modal problems.
- Incorporation of different game strategies – Pareto, Nash, Stackelberg
- Implementation of multi-fidelity approaches
- Parallel Computations
- Asynchronous evaluations

Further extensions of the Aircraft Simulation Model, and the developing and conducting of flight experiments with an UAS are presently under investigation.

Acknowledgments

The authors gratefully acknowledge Mourad Sefrioui, Dassault Aviation and Eric Whitney, Boeing Australia for fruitful discussions on Hierarchical EAs. Additional thanks goes to Pillar Eng of QUT and Australian Research Centre for Aerospace Automation (ARCAA) for assistance in the implementation of the Flight Planner within the Aircraft Simulation Model.

References

1. Jane's Unmanned Aerial Vehicles and Targets [online database], URL: http://juav.janes.com/public/juav/index.shtml [cited March 2007].
2. Kroo, I., Altus, S., Braun, R., Gage, P., and Sobieski, I., "Multidisciplinary Optimisation Methods for Aircraft Preliminary Design," AIAA 94-4325, Fifth AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimisation, September 7-9, 1994, Panama City, Florida.
3. Gonzalez, L., Whitney, E., Srinivas, K., Armfield, S., and Périaux, J., "A Robust Evolutionary Technique for Coupled and Multidisciplinary Design Optimisation Problems in Aeronautics", Computational Fluid Dynamics Journal, Vol. 14, No. 2, 2005, pp. 142-153.
4. Obayashi, S., "Multidisciplinary Design Optimisation of Aircraft Wing Planform Based on Evolutionary Algorithms," Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on, Vol. 4, 1998, pp. 3148-3153.
5. Sefrioui, M., and Périaux, J., "A Hierarchical Genetic Algorithm Using Multiple Models for Optimisation," Parallel Problem Solving from Nature – PPSN VI, edited by M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo and H.-P. Schwefel, Springer, 2000, pp. 879-888.
6. Hansen, N., and Ostermeier, A., "Completely Derandomized Self-Adaptation in Evolution Strategies," Evolutionary Computation, Vol. 9, No. 2, 2001, pp. 159-195.
7. Wakunda, J., and Zell, A., "Median-Selection for Parallel Steady-State Evolution Strategies," Parallel Problem Solving from Nature – PPSN VI, edited by M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo and H.-P. Schwefel, Springer, New York, 2000, pp. 879-888.
8. Michalewicz, Z., Genetic Algorithms + Data Structures = Evolution Programs, Artificial Intelligence, Springer-Verlag, New York, 1992.
9. Van Veldhuizen, D. A., Zydallis, J. B., and Lamont, G. B., "Considerations in Engineering Parallel Multiobjective Evolutionary Algorithms," IEEE Transactions on Evolutionary Computation, Vol. 7, No. 2, 2003, pp. 144-173.
10. Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., and Sunderam, V., "PVM: Parallel Virtual Mahine. A User's Guide and Tutorial for Networked Parallel Computing", Massachusetts Institute of Technology, Cambridge, MA, 1994.
11. AeroSim Blockset, Aeronautical Simulation Blockset, Ver. 1.2, Unmanned Dynamics, Hood River, OR, 2005.
12. McManus, I., Clothier, R., and Walker, R., "Highly Autonomous UAV Mission Planning and Piloting for Civilian Airspace Operations," Proceedings of the Eleventh Australian International Aerospace Congress, AIAC-11, First Australian Unmanned Air Vehicles Conference, 2005, Melbourne, Australia.
13. Kayton, M., "The Navigation Equations," Avionics Navigations Systems, edited by M. Kayton and W. R. Fried, John Wiley & Sons, New York, 1997, pp. 21-54.
14. Sinnot, R. W., "Virtues of the Haversine," Sky and Telescope, Vol. 68, No. 2, 1984, p. 159.
15. Biggs, M. C., "Constrained Minimization Using Recursive Quadratic Programming," Towards Global Optimization, edited by Dixon, L. C. W., and Szergo, G. P., North-Holland, Netherlands, 1975, pp. 341-349.
16. Han, S. P., "A Globally Convergent Method for Nonlinear Programming," Journal of Optimization Theory and Applications, Vol. 22, 1977, p. 297.
17. Powell, M. J. D., "The Convergence of Variable Metric Methods for Nonlinearly Constrained Optimization Calculations," Nonlinear Programming 3, edited by Mangasarian, O. L., Meyer, R. R., and Robinson, S. M., Academic Press, 1978.
18. Powell, M. J. D., "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations," Lecture Notes in Mathematics, edited by Watson, G. A., Vol. 630, Springer-Verlag, New York, 1978.
19. González, L.F, Whitney, E., Périaux, J., Sefrioui, S. and Srinivas, K. A Robust Evolutionary Technique for Inverse Aerodynamic Design Design and Control of Aerospace Systems Using Tools from Nature. Proceedings of the 4th European Congress on Computational Methods in Applied Sciences and Engineering, Volume II, ECCOMAS 2004, Jyvaskyla, Finland, July 24-28, 2004 editor: P. Neittaanmaki and T. Rossi and S. Korotov and E. Onate and J. Périaux and D. Knorzer, University of Jyvaskyla , Jyvaskyla , 2004 pages: CD ISBN 951-39-1869-6.